

U.S. NAVAL ACADEMY
COMPUTER SCIENCE DEPARTMENT
TECHNICAL REPORT



Evaluating MLNs for Collective Classification

Crane, Robert J.

USNA-CS-TR-2010-04

December 13, 2010

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 13 DEC 2010		2. REPORT TYPE		3. DATES COVERED 00-00-2010 to 00-00-2010	
4. TITLE AND SUBTITLE Evaluating MLNs for Collective Classification				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Naval Academy, Department Computer Science, 572M Holloway Rd, Annapolis, MD, 21402				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 21	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

Evaluating MLNs for Collective Classification

Robert J. Crane

Dept. Computer Science, U.S. Naval Academy
572M Holloway Rd, Annapolis, MD 21402

bob.j.crane@gmail.com

Abstract

Collective Classification is the process of labeling instances in a graph using both instance attribute information and information about relations between instances. While several Collective Classification Algorithms have been well studied, the use of Markov Logic Networks (MLNs) remains largely untested. MLNs pair first order logic statements with a numerical weight. With properly assigned weights, these rules may be used to infer class labels from evidence stated as logic statements. Our study evaluated MLNs against other Collective Classification algorithms on both synthetic data and real data from the CiteSeer dataset. As a whole, we encountered inconsistent and often poor performance with MLNs, especially on synthetic data where other Collective Classification algorithms performed well.

1 Introduction

Classification is the task of assigning appropriate labels to instances. These labels may represent categories or binary attributes of the instance and could represent any number of useful pieces of real information. A simple binary classification task may involve deciding if a webpage is purely advertisement or not. A more complicated task may involve categorizing emails into “work”, “family”, or “school” categories. Common classification techniques usually assume that data is drawn independently and identically (i.i.d.) from a distribution, in a manner where no correlation exists between instances.

Not only is this assumption often inaccurate, it precludes a large amount of data which is relational by design. Collective Classification proposes to take advantage of this feature by including relational information in the body of evidence used to assign labels. For instance, hyper-linked web-pages or scholarly articles with citations can be modeled as data with both independent attributes and relational data (i.e. links to other web-pages/articles). This technique has been shown to outperform traditional, independent clas-

sifiers in cases where instances, or “nodes”, of the same class link strongly to one another, a property called autocorrelation [12]. The difference in performance can be especially noticeable when some nodes in the test set have known labels [6].

Because independent classifiers have existed longer, they are much better studied than collective techniques. However a number of algorithms for Collective Classification have been proposed including relaxation labeling [1], iterative convergence techniques [12], belief propagation [16], and Gibbs sampling [6]. One relatively unexplored technique for Collective Classification involves the use of Markov Logic Networks (MLNs). A Markov Logic Network describes a series of first-order logic statements attached to numerical weights. Applied to Collective Classification, the logic statements of the MLN correlate the relational and independent features of datasets to the class label. A supervised learning technique is then used to determine the associated weights for each rule by training on a data set, the “training set,” with known labels. The MLN may then be used to infer over a data set, the “testing set,” with unknown labels and make predictions.

Our research compares techniques utilizing MLNs with other collective and independent classifiers. We describe the effect of utilizing different methods of weight learning and the results of different inference algorithms. We describe performance over both real and synthetic data. In the next section, we introduce MLNs, the associated weight-learning techniques, and the associated inference algorithms. In addition, we introduce the benchmark Collective Classification techniques with which we compare the use of MLNs. Next we describe the various conditions in which we tested MLNs and describe the results. Finally, we conclude and describe related research.

2 Background

While previous classification techniques utilize only independent instance attributes (usually, just “attributes”), Collective Classification takes advantage of the relation-

ships between instances. Because many datasets can be modeled this way, where relationships can be modeled as links between instances, Collective Classification has wide applicability and can outperform more traditional, attribute-only methods [11]. Classification of pages within a website is a clear example of Collective Classification. Attributes may include features like the presence of certain words while hyperlinks to other pages would provide relational attributes or “links”. Classifying a university department’s website may include labels like “Student Page”, “Instructor Page”, “Course Page”, etc.

Our study included a number of benchmark algorithms. The following sections introduce these algorithms as well as our techniques with MLNs.

2.1 Benchmark Algorithms

The *Iterative Classification Algorithm* (ICA) is a Collective Classification algorithm that begins with a bootstrapping process which computes initial predictions using only independent attributes. It then iteratively recomputes each node’s label using the predicted labels of neighboring nodes as evidence. It successively recomputes the unknown labels for a set number of iterations. In this form of the algorithm, the contribution of relational information from each neighboring node is weighted equally in class prediction [11]. In contrast, in the “cautious” variant relational information is used more heavily if the neighboring node’s label is considered more reliable [10].

Gibbs sampling is a well-studied Monte Carlo technique. At each state, it samples a label for each node based on the current predicted label distribution. The most likely label for each node is the one most often selected. The technique is shown to usually have good performance [10]. For this study, we chose ICA and Gibbs as representative of the “non-cautious” and “cautious” classes of algorithms described by McDowell et al. [10].

Relational Bayesian Classifier (RBC) is a non-collective algorithm. It represents heterogeneous data in such a manner that a Simple Bayesian Classifier may be used to learn conditional probabilities for each attribute. Our implementation uses a naive Bayes classifier, which assumes conditional independence between features.

The next algorithm, MRW, is an example of a relational-only classifier. It considers only relational information to label instances. Because it has no independent features to initiate the bootstrapping process, like Gibbs and ICA, this algorithm requires nodes with known groundings to be included in the testing set.

The *Multi-Rank Walk* (MRW) algorithm simulates random walks along links from groundings of each class and tallies the number of visits each node receives. Predictions are created from the relative number of visits each node re-

ceives from the candidate classes. It outperforms wvRN, a well known relational-only classifier, in certain situations, notably when the proportion of known labels is low [8].

2.2 MLNs

A *Markov Logic Network* (MLN) weds first-order logic with a statistically learned weight. To use an MLN, one must first create a set of first order logic rules. Weights are then attached to these rules by training on a set of data. There are various weight-learning techniques available although the Alchemy toolkit provides implementation for only generative learning, based on pseudo log-likelihood, and discriminative learning, based on Voted Perceptron, Conjugate Gradient, and Newton’s Method. With a set of rules with attached weights, the MLN may be used to infer probabilities for statements.

In the case of Collective Classification, links, attributes, and labels are represented in data sets as atomic formulas. Rules will take the form of first-order logic statements where attribute values imply certain labels and links between nodes imply the same label between these nodes. A weight-learning technique is then used to associate weights with these implication statements. These learned rules may then be used by inference algorithms, namely Markov Chain Monte Carlo (MCMC), Maximum-a-Posteriori (MAP), belief propagation [14], and MC-SAT [13] to infer probabilities for unlabeled nodes. It should be noted that the MCMC algorithm is the same basic algorithm as Gibbs except that the local classifier used to produce node predictions is the MLN instead of Naive Bayes.

3 Methods

3.1 Data Generation

Synthetic data produced with the Proximity toolkit provided us a robust source of material on which to test. Experiments on synthetic data included two training sets of 250 nodes and a single, 250 node testing set. In all experiments, we used 10 independent attributes and ran 10 trials using the same default settings for data generation as McDowell et al. [10].

Real data provided by the CiteSeer dataset provided another source of testing material. Again, we generated data using the same technique as McDowell et al. [10], a 5-fold cross validation, whereby 5 graphs of size 400 were generated with 4 used for training and the last for testing.

In our synthetic datasets we primarily modified the strength of the links as a predictor of like labels (the “homophily”), the strength of attributes as a predictor of label (the “attribute predictiveness”), the number of possible labels, and finally the proportion of test set nodes with known

labels (the “labeled proportion” or percentage of “groundings”). For our real data, we modified the number of independent attributes and the labeled proportion. The effects of these modifications on prediction accuracy is described in the results section.

3.2 Benchmark Algorithms

In order to evaluate the use of MLNs in Collective Classification, we utilized several other algorithms as points of comparison. These algorithms were implemented in the Proximity tool and were run on the same dataset as our MLN experiments. Specifically, we used the Relational Bayesian Classifier (RBC), ICA, and the Proximity implementation of Gibbs sampling as points of comparison. In addition to these algorithms, we implemented the relational-only classifier Multi-Rank Walk, shown in several circumstances to outperform the relational-only wvRN algorithm [7]. Each MLN trial ran the set of benchmark algorithms once and these results were averaged along with other trials of the same configuration. (For the synthetic data experiments, we ran ten trials; for the CiteSeer data, we ran five.)

3.3 MLN Comparison

MLN experiments utilized the Alchemy toolkit. In order to make appropriate comparisons, we exported data from the Proximity toolkit and created data files expressing attribute, label, and relational information as first order logic statements. Weight-learning was accomplished generatively or discriminatively. For discriminative learning, the class label predicate was specified as non-evidence. For generative learning, no non-evidence predicates were specified. The relative performance of these options motivated these choices. Inference was accomplished using four algorithms provided in the Alchemy toolkit: Maximum a posteriori (MAP), Markov chain Monte Carlo Gibbs Sampling (MCMC), MC-SAT(Poon and Domingos 2006), and belief propagation.

Each independent attribute j in the MLN had its own rule and was written in the form:

$$attr_j(object, +value) \Rightarrow class(object, +class)$$

The choice of rules introducing relational dependencies in the MLN proved to be very challenging. Ultimately, we found the single rule:

$$class(object1, class) \wedge LinkTo(object1, object2) \\ \Rightarrow class(object2, class)$$

to be the most successful. We attempted several variations using the $+$ operator attached to the class variable to imply potential dependencies between dissimilar classes but

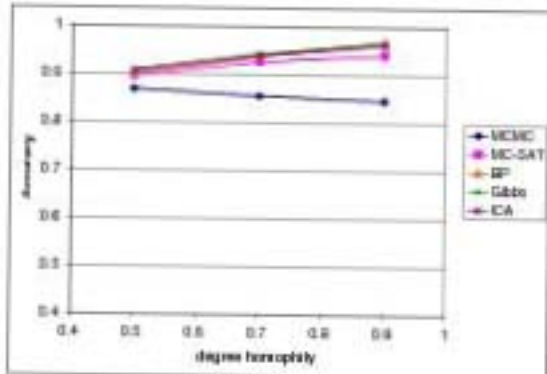


Figure 1. Shows performance of tested algorithms on binary data with 10% groundings. All MLN algorithms used a discriminatively learned MLN.

found this consistently hurt performance. Efforts to rewrite this relational rule as logically equivalent statements only produced identical performance.

For algorithms that provided a probability distribution over the labels, we simply chose the highest one and compared this set of labels to true labels to provide a measure of accuracy.

4 Results

Figures 1-3 highlight a few results comparing the average accuracy of the MLN algorithms and the benchmark algorithms as either homophily or attribute predictiveness/number is varied. As a whole, MLNs achieved lower accuracy than the benchmark algorithms, with a few exceptions. Tables containing detailed results are provided in the appendix. Below, we introduce a few themes from the results and propose a possible explanation for the inconsistent behavior of MLNs.

Discriminative learning was generally better than generative learning for MLNs. In almost all cases, discriminative weight learning yielded superior results to generative learning. This difference is especially pronounced for data with five class labels versus two labels. For example, Table 8 shows testing with five labels, 0% grounding. In this case, generative learning provided MLN inference algorithms little improvement to random guessing. A notable exception to this trend is exhibited in tests on binary data. In cases with high homophily, the MCMC inference algorithm using a generatively learned MLN yields consistently higher accuracy scores. Tables 1-6 contain these results. The 50% groundings case of the five class data (Table 12) also exhibits this result. Figure 1 displays another exception—the MCMC algorithm actually suffered from higher homophily

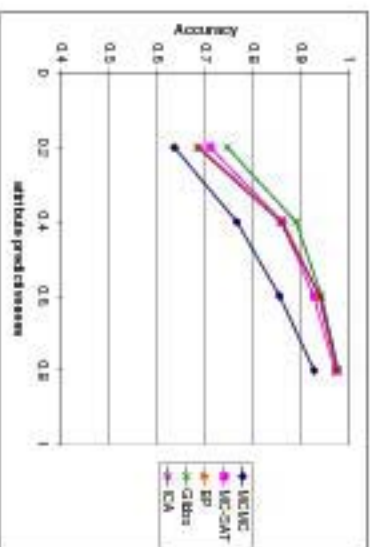


Figure 2. Shows performance on a set of binary data with 10% groundings. All MLN algorithms used a discriminatively learned MLN. The inference accuracy on binary data tended to be much better.

when learning discriminatively. The generative technique (results in Table 4) improves inference accuracy, as should be expected with higher homophily. However, besides these exceptions, discriminative learning was superior.

MLNs performed comparably much better with binary class labels than five class labels. One trait universal to all of our results was the superior performance of MLNs inferring on binary class label data. Not only is the binary classification more accurate in absolute terms, the performance of MLNs more closely rivals benchmark performance. While MLNs suffer particularly from 0% groundings on five class data, they perform much closer to benchmark algorithms on binary data. For instance, Figure 2 shows performance of MLN and benchmark algorithms on a set of binary data with 10% groundings. The performance of two of the MLN techniques, Belief Propagation and MC-SAT, is very close to the benchmark algorithms. Like most cases, Gibbs still outperformed the MLN algorithms.

The belief propagation algorithm behaved somewhat unpredictably. We note that it attempted to converge on a solution in a set number of iterations and that there was a marked difference in performance in cases where the algorithm successfully converged within the limited number of iterations.

MC-SAT outperforms others for 5 class labels and CiteSeer data. For 5 class labels the MC-SAT algorithm outperformed the other MLN algorithms in almost every case. For example, in the case of 10% groundings (shown in Table 9), MC-SAT outperforms its closest rival, belief propagation, in 9 of 12 cases. Nevertheless, even MC-SAT stills generally falls short of the benchmark algorithms. In the case of low homophily, attribute strength, and 0 or 10% labeled proportion, MC-SAT barely outperforms ICA. With 50% groundings, this slight advantage is crushed by ICA.

Interestingly, in the case of the real data, MC-SAT beats

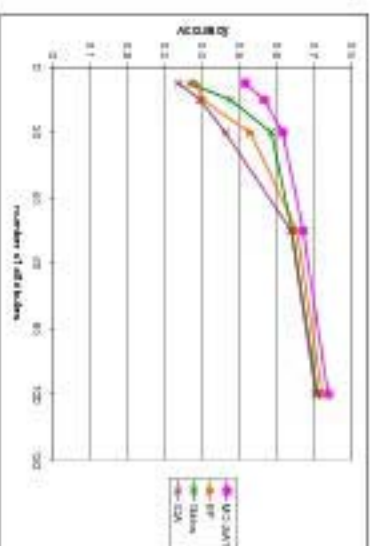


Figure 3. Shows performance of tested algorithms on CiteSeer data with 10% groundings. All MLN algorithms used a discriminatively learned MLN. This is the only case where an MLN inference algorithm performed consistently better than a benchmark.

the benchmark algorithms in every case. MC-SAT, using a discriminatively learned MLN, and performing on CiteSeer data is the only case in our study where an MLN algorithm consistently outperformed benchmarks. In every case of varied number of attributes and the labeled proportion, MC-SAT performed the best. Figure 3 shows this trend on a set of CiteSeer data with 10% groundings; the complete results can be found in Table 13. Below we introduce a possible explanation for this performance.

5 Discussion

For the synthetic data, the MLN algorithms consistently underperformed the benchmark algorithms. For the real data, however, one MLN algorithm (MC-SAT) performed noticeably better than all other algorithms. We considered whether this disparity might be due to the different size of the training data for the synthetic vs. the real data.

The real data used a training set of 1600 nodes while the synthetic data produced only sets of 500 nodes. To test the theory that MLN learning required a larger training set to be successful, we produced synthetic data with larger training sets and found that MLN classification did, in fact, improve. However, neither collective benchmark algorithm demonstrated any improvement with larger training sets. With our original training size, moderate homophily, and low attribute strength, our MLN algorithms rivaled ICA but fell short of Gibbs. When we increased the training set to 1000 nodes, inference accuracy for the MLNs only improved slightly and not enough to account for the disparity between CiteSeer and synthetic data.

6 Related Work

Collective Classification has been accomplished with a variety of algorithms. McDowell et al. [10] evaluated a variant of ICA, "Cautious ICA", which exploits more certain relational information to classify. Introducing this complexity improved the performance of ICA in most cases. When tested on the CiteSeer database, Cautious ICA generally outperformed its non-cautious rivals, especially when using fewer non-relational attributes.

Dhurandhar and Dobra compared the performance of MLNs against Relational Dependency Networks [3]. They, however, only utilized MCMC and MAP inference, using both Generative and Discriminative weight-learning. Unlike us, they found that the choice of weight-learning technique and inference algorithm did not qualitatively affect the results. In particular, they found that Relational Dependency Networks with Gibbs performed comparably to MLNs. Our data, however, showed consistent underperformance of MLNs compared to Gibbs. Their use of data with mainly binary class labels may explain this difference, as our results indicate that binary labels provide for more comparable performance for MLNs.

Markov Logic Networks provide a tool with a diverse range of applications. Chechotka, et al. [2] utilize MLNs to collectively classify entities identified in images. Relational information is defined as attributes shared commonly between entities in different pictures. Unlike our data, this provides multiple link types and, as they found, complicates classification since different links may vary in importance to classification. They too used the Alchemy toolkit, although they only used a single set of weight-learning and inference settings.

One area of MLN study receiving attention is the weight learning technique, an intractable problem with several candidate methods. Lowd and Domingos [9] explored alternatives which improve upon existing techniques by using second-order information or by modifying the learning rate for different clauses. Although they too used the Alchemy toolkit, implementing their techniques as extensions, they tested on real datasets we didn't use, Cora and WebKB, using the latter for Collective Classification. The algorithms they introduced improved accuracy over their metric implemented in Alchemy. However, they did not compare Collective Classification accuracy to methods outside MLNs. Huynh and Mooney [5] introduce a method of discriminative learning based on a max-margin framework which can optimize MLNs for collective classification accuracy. They too tested on the CiteSeer database as well as WebKB and utilized the Alchemy toolkit. The resulting learner provided equal or better performance than the existing discriminative learning techniques.

Finally, Markov Logic Networks and Collective Classi-

fication has applications outside of the networked data representations we used. Domingos and Richardson [4] describe link prediction, link-based clustering, social network modeling, and object identification in an MLN framework. Riedel and Meza-Ruiz use MLNs for natural language processing, taking advantage of relational aspects of semantics [15].

7 Future Work

The disparity between the results we found for real and synthetic data raises the question of how the nature of the data can affect performance. The data we fed into Alchemy, whether real or synthetic, had similar characteristics in terms of homophily and link density. Nevertheless, MLN performance was far superior with the real data. Further work should be done into how the nature of data, especially real data can affect the performance of MLNs.

As mentioned in the related work section, other methods of weight learning are being explored and, given our experience with dramatically different results given the two methods we tested, may yield performance increases that could improve current performance levels. Given the computational time required by the discriminative method, a more efficient learning technique that matched discriminative techniques in performance would go far to make MLNs a more viable approach to Collective Classification.

Finally, our experience with performance increases with a larger training sets begs more exploration into the amount of data needed to train MLNs.

8 Conclusion

As a Collective Classification tool, we found unpredictable results for MLNs. It is notable too that the computation time to learn the rule weights discriminatively was significant and dwarfed the total time required by the superior benchmark algorithms. For the synthetic data, performance was consistently poor with the techniques tested and unreliable across different data types, whether altering the number of labels or labeled proportion. However, given the performance disparity between the real and synthetic data, it is possible that the nature of our testing data affected the performance where other types of data may have yielded better results. Future work should explore these effects in more detail.

References

- [1] S. Chakrabarti, B. Dorn, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 307–318, 1998.

- [2] A. Chechotka, D. Dash, and M. Philipose. Relational learning for collective classification of entities in images. In *AAAI-10 Workshop on Statistical Relational AI*, 2010.
- [3] A. Dhurandhar and A. Dobra. Collective vs independent classification in statistical relational learning. In *Submitted for publication*, 2010.
- [4] P. Domingos and M. Richardson. Markov logic: A unifying framework for statistical relational learning. In *Proceedings of the ICML-2004 Workshop on Statistical Relational Learning and its connections to other fields*, pages 49–54, 2004.
- [5] T. N. Huynh and R. J. Mooney. Max-margin weight learning for markov logic networks. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part I, ECML PKDD '09*, pages 564–579, Berlin, Heidelberg, 2009. Springer-Verlag.
- [6] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 593–598, 2004.
- [7] F. Lin and W. Cohen. Semi-Supervised Classification of Network Data Using Very Few Labels. In *The International Conference on Advances in Social Network Analysis and Mining (ASONAM)*, 2010.
- [8] F. Lin and W. W. Cohen. Semi-supervised classification of network data using very few labels. In *Proceedings of*, 2010.
- [9] D. Lowd and P. Domingos. Efficient weight learning for markov logic networks. In *In Proceedings of the Eleventh European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 200–211, 2007.
- [10] L. McDowell, K. Gupta, and D. Aha. Cautious collective classification. *The Journal of Machine Learning Research*, 10:2777–2836, 2009.
- [11] L. McDowell, K. Gupta, and D. Aha. Meta-Prediction for Collective Classification. In *Proceedings of the Twenty-Third International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2010.
- [12] J. Neville and D. Jensen. Iterative classification in relational data. In *Proceedings of the Workshop on Learning Statistical Models from Relational Data at the 17th National Conference on Artificial Intelligence (AAAI)*, pages 13–20, 2000.
- [13] H. Poon and P. Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, pages 458–463. AAAI Press, 2006.
- [14] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.
- [15] S. Riedel and I. Meza-Ruiz. Collective semantic role labelling with markov logic. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning, CoNLL '08*, pages 193–197, Morristown, NJ, USA, 2008. Association for Computational Linguistics.
- [16] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 485–492, 2002.

Table 1. 2-Class, $lp=0\%$, Discriminative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.62	0.607	0.63
ap=0.4	0.76	0.754	0.772
ap=0.6	0.878	0.87	0.875
ap=0.8	0.945	0.949	0.948

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.617	0.625	0.706
ap=0.4	0.79	0.837	0.89
ap=0.6	0.902	0.932	0.958
ap=0.8	0.955	0.974	0.983

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.496	0.498	0.496
ap=0.4	0.496	0.498	0.496
ap=0.6	0.496	0.498	0.496
ap=0.8	0.496	0.498	0.496

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.617	0.631	0.745
ap=0.4	0.812	0.867	0.95
ap=0.6	0.903	0.938	0.966
ap=0.8	0.957	0.974	0.983

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.577	0.612	0.649
ap=0.4	0.766	0.797	0.862
ap=0.6	0.855	0.884	0.934
ap=0.8	0.897	0.925	0.969

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.6	0.595	0.585
ap=0.4	0.746	0.732	0.712
ap=0.6	0.864	0.84	0.814
ap=0.8	0.931	0.914	0.922

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.609	0.619	0.835
ap=0.4	0.804	0.868	0.953
ap=0.6	0.901	0.931	0.966
ap=0.8	0.954	0.97	0.979

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.618	0.62	0.707
ap=0.4	0.802	0.849	0.882
ap=0.6	0.899	0.921	0.943
ap=0.8	0.952	0.962	0.953

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 2. 2-Class, $lp=0\%$, Generative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.62	0.607	0.63
ap=0.4	0.76	0.754	0.772
ap=0.6	0.878	0.87	0.875
ap=0.8	0.945	0.949	0.948

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.617	0.625	0.706
ap=0.4	0.79	0.837	0.89
ap=0.6	0.902	0.932	0.958
ap=0.8	0.955	0.974	0.983

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.496	0.498	0.496
ap=0.4	0.496	0.498	0.496
ap=0.6	0.496	0.498	0.496
ap=0.8	0.496	0.498	0.496

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.617	0.631	0.745
ap=0.4	0.812	0.867	0.95
ap=0.6	0.903	0.938	0.966
ap=0.8	0.957	0.974	0.983

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.517	0.515	0.562
ap=0.4	0.66	0.682	0.804
ap=0.6	0.804	0.84	0.884
ap=0.8	0.88	0.91	0.951

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.579	0.522	0.642
ap=0.4	0.664	0.638	0.905
ap=0.6	0.815	0.773	0.933
ap=0.8	0.919	0.952	0.974

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.603	0.57	0.489
ap=0.4	0.781	0.823	0.912
ap=0.6	0.895	0.924	0.958
ap=0.8	0.956	0.968	0.981

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.531	0.579	0.526
ap=0.4	0.667	0.661	0.609
ap=0.6	0.8	0.766	0.691
ap=0.8	0.892	0.882	0.775

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 3. 2-Class, $I_p=10\%$, Discriminative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.624	0.611	0.626
ap=0.4	0.763	0.756	0.771
ap=0.6	0.88	0.874	0.874
ap=0.8	0.944	0.949	0.949

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.658	0.684	0.731
ap=0.4	0.804	0.856	0.9
ap=0.6	0.906	0.938	0.96
ap=0.8	0.956	0.974	0.983

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.627	0.752	0.925
ap=0.4	0.605	0.748	0.927
ap=0.6	0.622	0.757	0.927
ap=0.8	0.626	0.747	0.925

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.638	0.745	0.911
ap=0.4	0.819	0.888	0.955
ap=0.6	0.905	0.941	0.965
ap=0.8	0.959	0.976	0.983

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.625	0.693	0.77
ap=0.4	0.764	0.819	0.878
ap=0.6	0.858	0.896	0.94
ap=0.8	0.906	0.937	0.976

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.628	0.635	0.631
ap=0.4	0.757	0.765	0.739
ap=0.6	0.867	0.854	0.844
ap=0.8	0.932	0.926	0.924

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.673	0.687	0.832
ap=0.4	0.811	0.862	0.949
ap=0.6	0.9	0.936	0.966
ap=0.8	0.954	0.972	0.979

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.678	0.71	0.807
ap=0.4	0.818	0.859	0.898
ap=0.6	0.895	0.925	0.94
ap=0.8	0.953	0.969	0.958

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 4. 2-Class, $lp=10\%$, Generative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.624	0.611	0.626
ap=0.4	0.763	0.756	0.771
ap=0.6	0.88	0.874	0.874
ap=0.8	0.944	0.949	0.949

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.658	0.684	0.731
ap=0.4	0.804	0.856	0.9
ap=0.6	0.906	0.938	0.96
ap=0.8	0.956	0.974	0.983

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.627	0.752	0.925
ap=0.4	0.605	0.748	0.927
ap=0.6	0.622	0.757	0.927
ap=0.8	0.626	0.747	0.925

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.638	0.745	0.911
ap=0.4	0.819	0.888	0.955
ap=0.6	0.905	0.941	0.965
ap=0.8	0.959	0.976	0.983

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.545	0.62	0.761
ap=0.4	0.703	0.755	0.833
ap=0.6	0.819	0.853	0.9
ap=0.8	0.884	0.921	0.954

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.585	0.575	0.897
ap=0.4	0.697	0.672	0.917
ap=0.6	0.817	0.807	0.942
ap=0.8	0.928	0.955	0.977

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.623	0.629	0.729
ap=0.4	0.79	0.794	0.891
ap=0.6	0.898	0.925	0.955
ap=0.8	0.957	0.969	0.98

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.517	0.559	0.578
ap=0.4	0.655	0.619	0.637
ap=0.6	0.79	0.736	0.731
ap=0.8	0.892	0.869	0.819

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 5. 2-Class, $I_p=50\%$, Discriminative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.637	0.601	0.623
ap=0.4	0.771	0.758	0.76
ap=0.6	0.876	0.868	0.868
ap=0.8	0.941	0.946	0.948

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.743	0.819	0.905
ap=0.4	0.825	0.886	0.942
ap=0.6	0.911	0.938	0.968
ap=0.8	0.959	0.976	0.983

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.711	0.845	0.955
ap=0.4	0.717	0.822	0.949
ap=0.6	0.701	0.842	0.95
ap=0.8	0.706	0.834	0.958

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.755	0.867	0.954
ap=0.4	0.832	0.899	0.956
ap=0.6	0.906	0.941	0.967
ap=0.8	0.96	0.976	0.983

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.717	0.798	0.897
ap=0.4	0.805	0.874	0.941
ap=0.6	0.888	0.927	0.963
ap=0.8	0.923	0.964	0.978

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.721	0.778	0.864
ap=0.4	0.813	0.851	0.901
ap=0.6	0.891	0.915	0.948
ap=0.8	0.95	0.956	0.972

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.726	0.726	0.798
ap=0.4	0.822	0.855	0.928
ap=0.6	0.897	0.935	0.966
ap=0.8	0.956	0.972	0.978

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.748	0.842	0.918
ap=0.4	0.821	0.887	0.951
ap=0.6	0.899	0.933	0.967
ap=0.8	0.958	0.97	0.967

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 6. 2-Class, $lp=50\%$, Generative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.637	0.601	0.623
ap=0.4	0.771	0.758	0.76
ap=0.6	0.876	0.868	0.868
ap=0.8	0.941	0.946	0.948

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.743	0.819	0.905
ap=0.4	0.825	0.886	0.942
ap=0.6	0.911	0.938	0.968
ap=0.8	0.959	0.976	0.983

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.711	0.845	0.955
ap=0.4	0.717	0.822	0.949
ap=0.6	0.701	0.842	0.95
ap=0.8	0.706	0.834	0.958

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.755	0.867	0.954
ap=0.4	0.832	0.899	0.956
ap=0.6	0.906	0.941	0.967
ap=0.8	0.96	0.976	0.983

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.699	0.809	0.887
ap=0.4	0.749	0.831	0.908
ap=0.6	0.859	0.905	0.945
ap=0.8	0.916	0.965	0.975

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.692	0.813	0.947
ap=0.4	0.787	0.852	0.944
ap=0.6	0.876	0.921	0.951
ap=0.8	0.946	0.964	0.978

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.648	0.645	0.698
ap=0.4	0.781	0.781	0.805
ap=0.6	0.878	0.882	0.912
ap=0.8	0.949	0.963	0.98

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.658	0.699	0.764
ap=0.4	0.752	0.751	0.811
ap=0.6	0.841	0.835	0.879
ap=0.8	0.929	0.921	0.925

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 7. 5-Class, $lp=0\%$, Discriminative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.364	0.357	0.369
ap=0.4	0.491	0.49	0.502
ap=0.6	0.614	0.619	0.626
ap=0.8	0.727	0.731	0.733

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.353	0.365	0.379
ap=0.4	0.507	0.551	0.601
ap=0.6	0.645	0.734	0.785
ap=0.8	0.78	0.845	0.896

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.203	0.202	0.202
ap=0.4	0.203	0.202	0.202
ap=0.6	0.203	0.202	0.202
ap=0.8	0.203	0.202	0.202

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.39	0.479	0.504
ap=0.4	0.579	0.684	0.824
ap=0.6	0.713	0.82	0.92
ap=0.8	0.808	0.884	0.948

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.28	0.264	0.228
ap=0.4	0.348	0.376	0.366
ap=0.6	0.44	0.504	0.531
ap=0.8	0.572	0.624	0.646

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.338	0.291	0.246
ap=0.4	0.444	0.399	0.362
ap=0.6	0.544	0.519	0.482
ap=0.8	0.669	0.626	0.579

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.373	0.369	0.294
ap=0.4	0.512	0.528	0.545
ap=0.6	0.64	0.687	0.72
ap=0.8	0.797	0.838	0.856

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.368	0.303	0.28
ap=0.4	0.543	0.508	0.458
ap=0.6	0.663	0.722	0.678
ap=0.8	0.768	0.805	0.813

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 8. 5-Class, $lp=0\%$, Generative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.364	0.357	0.369
ap=0.4	0.491	0.49	0.502
ap=0.6	0.614	0.619	0.626
ap=0.8	0.727	0.731	0.733

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.353	0.365	0.379
ap=0.4	0.507	0.551	0.601
ap=0.6	0.645	0.734	0.785
ap=0.8	0.78	0.845	0.896

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.203	0.202	0.202
ap=0.4	0.203	0.202	0.202
ap=0.6	0.203	0.202	0.202
ap=0.8	0.203	0.202	0.202

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.39	0.479	0.504
ap=0.4	0.579	0.684	0.824
ap=0.6	0.713	0.82	0.92
ap=0.8	0.808	0.884	0.948

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.2	0.199	0.216
ap=0.4	0.196	0.202	0.205
ap=0.6	0.21	0.221	0.225
ap=0.8	0.234	0.252	0.256

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.198	0.196	0.196
ap=0.4	0.202	0.196	0.196
ap=0.6	0.211	0.197	0.196
ap=0.8	0.231	0.197	0.196

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.195	0.188	0.122
ap=0.4	0.195	0.188	0.123
ap=0.6	0.195	0.187	0.126
ap=0.8	0.201	0.199	0.135

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.204	0.196	0.223
ap=0.4	0.197	0.216	0.237
ap=0.6	0.198	0.214	0.221
ap=0.8	0.234	0.232	0.272

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 9. 5-Class, $Ip=10\%$, Discriminative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.364	0.357	0.37
ap=0.4	0.486	0.487	0.5
ap=0.6	0.612	0.616	0.627
ap=0.8	0.727	0.729	0.736

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.382	0.437	0.482
ap=0.4	0.531	0.611	0.664
ap=0.6	0.672	0.761	0.824
ap=0.8	0.789	0.85	0.918

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.428	0.6	0.833
ap=0.4	0.428	0.615	0.833
ap=0.6	0.425	0.603	0.841
ap=0.8	0.428	0.602	0.837

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.483	0.622	0.791
ap=0.4	0.623	0.761	0.889
ap=0.6	0.72	0.834	0.929
ap=0.8	0.813	0.883	0.951

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.293	0.327	0.285
ap=0.4	0.387	0.387	0.426
ap=0.6	0.471	0.518	0.572
ap=0.8	0.564	0.648	0.672

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.355	0.353	0.309
ap=0.4	0.473	0.463	0.443
ap=0.6	0.577	0.579	0.552
ap=0.8	0.683	0.67	0.655

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.386	0.401	0.344
ap=0.4	0.523	0.547	0.56
ap=0.6	0.642	0.695	0.745
ap=0.8	0.794	0.829	0.873

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.423	0.431	0.364
ap=0.4	0.58	0.63	0.571
ap=0.6	0.677	0.737	0.769
ap=0.8	0.792	0.829	0.816

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 10. 5-Class, $lp=10\%$, Generative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.364	0.357	0.37
ap=0.4	0.486	0.487	0.5
ap=0.6	0.612	0.616	0.627
ap=0.8	0.727	0.729	0.736

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.382	0.437	0.482
ap=0.4	0.531	0.611	0.664
ap=0.6	0.672	0.761	0.824
ap=0.8	0.789	0.85	0.918

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.428	0.6	0.833
ap=0.4	0.428	0.615	0.833
ap=0.6	0.425	0.603	0.841
ap=0.8	0.428	0.602	0.837

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.483	0.622	0.791
ap=0.4	0.623	0.761	0.889
ap=0.6	0.72	0.834	0.929
ap=0.8	0.813	0.883	0.951

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.226	0.248	0.389
ap=0.4	0.231	0.266	0.375
ap=0.6	0.249	0.286	0.386
ap=0.8	0.271	0.33	0.445

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.241	0.255	0.367
ap=0.4	0.244	0.257	0.351
ap=0.6	0.252	0.265	0.388
ap=0.8	0.274	0.288	0.486

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.195	0.196	0.254
ap=0.4	0.197	0.196	0.259
ap=0.6	0.201	0.219	0.269
ap=0.8	0.218	0.252	0.29

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.284	0.311	0.321
ap=0.4	0.279	0.297	0.332
ap=0.6	0.293	0.331	0.37
ap=0.8	0.306	0.352	0.402

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 11. 5-Class, $lp=50\%$, Discriminative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.355	0.359	0.378
ap=0.4	0.473	0.491	0.524
ap=0.6	0.62	0.616	0.637
ap=0.8	0.735	0.724	0.752

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.574	0.705	0.861
ap=0.4	0.65	0.784	0.901
ap=0.6	0.73	0.831	0.933
ap=0.8	0.802	0.891	0.95

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.577	0.763	0.912
ap=0.4	0.571	0.746	0.911
ap=0.6	0.568	0.765	0.909
ap=0.8	0.578	0.76	0.906

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.613	0.798	0.915
ap=0.4	0.664	0.823	0.925
ap=0.6	0.737	0.856	0.944
ap=0.8	0.808	0.905	0.961

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.389	0.399	0.421
ap=0.4	0.463	0.472	0.534
ap=0.6	0.525	0.585	0.617
ap=0.8	0.617	0.693	0.743

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.553	0.625	0.644
ap=0.4	0.629	0.694	0.78
ap=0.6	0.72	0.757	0.841
ap=0.8	0.798	0.823	0.889

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.459	0.499	0.474
ap=0.4	0.577	0.612	0.655
ap=0.6	0.672	0.711	0.756
ap=0.8	0.786	0.821	0.883

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.566	0.662	0.578
ap=0.4	0.656	0.747	0.82
ap=0.6	0.739	0.791	0.864
ap=0.8	0.806	0.875	0.909

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 12. 5-Class, $lp=50\%$, Generative

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.355	0.359	0.378
ap=0.4	0.473	0.491	0.524
ap=0.6	0.62	0.616	0.637
ap=0.8	0.735	0.724	0.752

RBC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.574	0.705	0.861
ap=0.4	0.65	0.784	0.901
ap=0.6	0.73	0.831	0.933
ap=0.8	0.802	0.891	0.95

ICA

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.577	0.763	0.912
ap=0.4	0.571	0.746	0.911
ap=0.6	0.568	0.765	0.909
ap=0.8	0.578	0.76	0.906

MRW

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.613	0.798	0.915
ap=0.4	0.664	0.823	0.925
ap=0.6	0.737	0.856	0.944
ap=0.8	0.808	0.905	0.961

Proximity Gibbs

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.387	0.485	0.699
ap=0.4	0.393	0.517	0.73
ap=0.6	0.386	0.573	0.737
ap=0.8	0.462	0.624	0.761

MAP

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.478	0.62	0.881
ap=0.4	0.485	0.634	0.876
ap=0.6	0.497	0.681	0.891
ap=0.8	0.519	0.717	0.899

MCMC

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.207	0.227	0.625
ap=0.4	0.219	0.235	0.667
ap=0.6	0.237	0.266	0.74
ap=0.8	0.277	0.346	0.803

Belief Propagation

	dh=0.5	dh=0.7	dh=0.9
ap=0.2	0.418	0.548	0.619
ap=0.4	0.428	0.555	0.634
ap=0.6	0.43	0.549	0.652
ap=0.8	0.464	0.575	0.646

MC-SAT

Average accuracy results for various algorithms; dh is the measure of homophily; ap is the measure of attribute predictiveness

Table 13. Discriminative Learning on CiteSeer

	lp=0%	lp=10%	lp=50%
nAttr=5	0.297	0.295	0.291
nAttr=10	0.332	0.333	0.337
nAttr=20	0.364	0.364	0.353
nAttr=50	0.539	0.541	0.54
nAttr=100	0.638	0.638	0.631

RBC

	lp=0%	lp=10%	lp=50%
nAttr=5	0.298	0.336	0.567
nAttr=10	0.34	0.394	0.619
nAttr=20	0.401	0.461	0.633
nAttr=50	0.61	0.638	0.685
nAttr=100	0.696	0.704	0.741

ICA

	lp=0%	lp=10%	lp=50%
nAttr=5	0.058	0.624	0.654
nAttr=10	0.058	0.623	0.66
nAttr=20	0.058	0.625	0.663
nAttr=50	0.058	0.626	0.666
nAttr=100	0.058	0.622	0.657

MRW

	lp=0%	lp=10%	lp=50%
nAttr=5	0.243	0.366	0.653
nAttr=10	0.368	0.475	0.674
nAttr=20	0.531	0.585	0.706
nAttr=50	0.602	0.64	0.702
nAttr=100	0.688	0.709	0.748

Proximity Gibbs

	lp=0%	lp=10%	lp=50%
nAttr=5	0.19	0.208	0.251
nAttr=10	0.2	0.234	0.274
nAttr=20	0.292	0.293	0.314
nAttr=50	0.36	0.376	0.372
nAttr=100	0.419	0.408	0.413

MAP

	lp=0%	lp=10%	lp=50%
nAttr=5	0.221	0.288	0.519
nAttr=10	0.241	0.308	0.54
nAttr=20	0.35	0.409	0.572
nAttr=50	0.529	0.567	0.651
nAttr=100	0.65	0.67	0.724

MCMC

	lp=0%	lp=10%	lp=50%
nAttr=5	0.316	0.383	0.567
nAttr=10	0.345	0.4	0.557
nAttr=20	0.511	0.528	0.591
nAttr=50	0.644	0.653	0.657
nAttr=100	0.714	0.721	0.733

Belief Propagation

	lp=0%	lp=10%	lp=50%
nAttr=5	0.325	0.515	0.652
nAttr=10	0.443	0.565	0.695
nAttr=20	0.545	0.615	0.682
nAttr=50	0.672	0.67	0.717
nAttr=100	0.725	0.737	0.742

MC-SAT

Average accuracy results for various algorithms; nAttr is the number of independent attributes used; lp is the labeled proportion

Table 14. Generative Learning on CiteSeer

	lp=0%	lp=10%	lp=50%
nAttr=5	0.297	0.295	0.291
nAttr=10	0.332	0.333	0.337
nAttr=20	0.364	0.364	0.353
nAttr=50	0.539	0.541	0.54
nAttr=100	0.638	0.638	0.631

RBC

	lp=0%	lp=10%	lp=50%
nAttr=5	0.298	0.336	0.567
nAttr=10	0.34	0.394	0.619
nAttr=20	0.401	0.461	0.633
nAttr=50	0.61	0.638	0.685
nAttr=100	0.696	0.704	0.741

ICA

	lp=0%	lp=10%	lp=50%
nAttr=5	0.058	0.624	0.654
nAttr=10	0.058	0.623	0.66
nAttr=20	0.058	0.625	0.663
nAttr=50	0.058	0.626	0.666
nAttr=100	0.058	0.622	0.657

MRW

	lp=0%	lp=10%	lp=50%
nAttr=5	0.243	0.366	0.653
nAttr=10	0.368	0.475	0.674
nAttr=20	0.531	0.585	0.706
nAttr=50	0.602	0.64	0.702
nAttr=100	0.688	0.709	0.748

Proximity Gibbs

	lp=0%	lp=10%	lp=50%
nAttr=5	0.169	0.213	0.338
nAttr=10	0.241	0.236	0.299
nAttr=20	0.227	0.247	0.28
nAttr=50	0.298	0.314	0.314
nAttr=100	0.339	0.344	0.35

MAP

	lp=0%	lp=10%	lp=50%
nAttr=5	0.209	0.263	0.473
nAttr=10	0.207	0.241	0.398
nAttr=20	0.239	0.273	0.394
nAttr=50	0.289	0.337	0.46
nAttr=100	0.313	0.364	0.447

MCMC

	lp=0%	lp=10%	lp=50%
nAttr=5	0.2	0.216	0.33
nAttr=10	0.199	0.203	0.254
nAttr=20	0.238	0.248	0.279
nAttr=50	0.335	0.342	0.372
nAttr=100	0.361	0.363	0.362

Belief Propagation

	lp=0%	lp=10%	lp=50%
nAttr=5	0.198	0.291	0.541
nAttr=10	0.211	0.278	0.408
nAttr=20	0.278	0.314	0.419
nAttr=50	0.343	0.376	0.48
nAttr=100	0.372	0.409	0.441

MC-SAT

Average accuracy results for various algorithms; nAttr is the number of independent attributes used; lp is the labeled proportion